Testing Graph Blow-Up*

Lidor Avigad†
78 Arlozorov strert
Rehovot
ISRAEL
avigadl@gmail.com

Oded Goldreich‡
Department of Computer Science
Weizmann Institute of Science
Rehovot, ISRAEL.
oded.goldreich@weizmann.ac.il

March 15, 2010

Abstract

Referring to the query complexity of testing graph properties in the adjacency matrix model, we advance the study of the class of properties that can be tested non-adaptively within complexity that is inversely proportional to the proximity parameter. Arguably, this is the lowest meaningful complexity class in this model, and we show that it contains a very natural class of graph properties. Specifically, for every fixed graph H, we consider the set of all graphs that are obtained by a (possibly unbalanced) blow-up of H. We show a non-adaptive tester of query complexity $\widetilde{O}(1/\epsilon)$ that distinguishes graphs that are a blow-up of H from graphs that are ϵ-far from any such blow-up.

Keywords: Property Testing, Adaptivity vs Non-adaptivity, One-sided vs Two-sided Error, Graph Properties, Graph Blow-up

*This work is based on the M.Sc. thesis of the first author [A], which was completed under the supervision of the second author.

†Research performed while the author was a graduate student at the Weizmann Institute of Science.

‡Partially supported by the Israel Science Foundation (grant No. 1041/08).

1 Introduction

The general context of this work is that of testing graph properties in the adjacency matrix representation (as initiated in [GGR]). In this model graphs are viewed as (symmetric) Boolean functions over a domain consisting of all possible vertex-pairs (i.e., an N-vertex graph $G = ([N], E)$ is represented by the function $g : [N] \times [N] \to \{0,1\}$ such that $\{u,v\} \in E$ if and only if $g(u,v) = 1$). Consequently, an N-vertex graph represented by the function $g : [N] \times [N] \to \{0,1\}$ is said to be ϵ-far from some predetermined graph property if more than $\epsilon \cdot N^2$ entries of g must be modified in order to yield a representation of a graph that has this property. We refer to ϵ as the proximity parameter, and the complexity of testing is stated in terms of ϵ and the number of vertices in the graph (i.e., N).

Interestingly, many natural graph properties can be tested within query complexity that depends only on the proximity parameter; see [GGR], which presents testers with query complexity $\text{poly}(1/\epsilon)$, and [AFNS], which characterizes the class of properties that are testable within query complexity that depends only on the proximity parameter (where this dependence may be an arbitrary function of ϵ). A well-known open problem in this area is to characterize the class of graph properties that can be tested within query complexity $\text{poly}(1/\epsilon)$. We mention that such a characterization has been obtained in the special case of induced subgraph freeness properties [AS], but the general case seems quite difficult.

In light of this state of affairs, it was suggested in [GR08] to try to characterize lower query complexity classes, and in particular the class of graph properties that can be tested non-adaptively within query complexity $\widetilde{O}(1/\epsilon)$. As a first step towards this goal, it was shown in [GR08, Sec. 6] that, for every constant c, the set of graphs that each consists of at most c isolated cliques is such a property.

In this work we significantly extend the latter result by showing that the class of graph properties that can be tested non-adaptively within query complexity $\widetilde{O}(1/\epsilon)$ contains all graph blow-up properties. For any fixed graph $H = ([h], F)$, we say that a graph $G = ([N], E)$ is a blow-up of H if the vertices of G can be clustered in up to h clusters such that the edges between these clusters reflect the edge relation of H. That is, vertices in the i^{th} and j^{th} cluster are connected in G if and only if $(i,j) \in F$. Note that, unlike in the case of balanced blow-up (cf. [GKNR]), the clusters are not required to have equal size.[1] Also note that the "collection of c cliques" property studied in [GR08, Sec. 6] can be cast as the property of being a blow-up of a c-vertex clique (by considering the complement graph).

Theorem 1.1 (main result): *For every fixed H, the property of being a blow-up of H is testable by $\widetilde{O}(1/\epsilon)$ non-adaptive queries. Furthermore, the tester has one-sided error* (i.e., it always accepts graphs that are blow-ups of H) *and runs in* $\text{poly}(1/\epsilon)$*-time.*

We mention that, except for $h = 1$, the aforementioned property cannot be tested by $o(1/\epsilon)$ queries, even when adaptivity and two-sided error are allowed (see [GR08, Prop. 6.1]). We also mention that, by [GR08, Prop. 6.2], a tester of $\widetilde{O}(1/\epsilon)$ query complexity cannot be canonical (i.e., it cannot rule by inspecting an induced subgraph).

Additional results. We also consider the complexity of testing "balanced blow-up" properties, showing that the two-sided error query complexity is quadratic in $1/\epsilon$ for both adaptive and non-adaptive testers; see Proposition 2.4. Finally, we present proximity oblivious testers (cf. [GR09]) for any (general) blow-up property; see Theorem 5.2.

[1] We note that testing balanced blow-up properties requires $\Omega(1/\epsilon^2)$ queries. For details, see Section 2.2.

Techniques. Theorem 1.1 is proved by presenting a suitable tester and analyzing it. Recall that this tester cannot be canonical; indeed, this tester selects at random a sample of $\widetilde{O}(1/\epsilon)$ vertices, but it inspects (or queries) only $\widetilde{O}(1/\epsilon)$ of the vertex pairs in this sample. Consequently, the tester (and the analysis) has to deal with partial knowledge of the subgraph induced by the sample. A pivotal notion regarding such partial views is of "inconsistency" between vertices (w.r.t a given partial view), which means that these vertices have different neighbor sets and thus cannot be placed in the same cluster (of a blow-up of H (or any other graph)). Specifically, the tester considers all sets of up to $h+1$ pairwise inconsistent vertices, and accepts if and only if each such set (along with the known incidence relations) can be embedded in H. As usual, the technically challenging part is analyzing the behavior of the tester on arbitrary graphs that are far from being blow-ups of H. Our analysis proceeds in iterations, where in each iteration some progress is made, but this progress is not reflected by a growing number of incidence contraints but rather in the decreasing density of the violations reflected in the incidence constraints. This progress is captured in Lemma 4.4 (which refers to notions introduced in Section 4.1). Here we merely mention that the number of iterations is polylogarithmic in ϵ^{-1} rather than being $O(h^2)$.

Organization. The core of this paper is presented in Sections 3 and 4, which contain a description of the tester and its analysis, respectively. (Indeed, this part establishes Theorem 1.1.) Section 2 provides preliminaries, which may be skipped by the experts, as well as a side discussion (and result) regarding "balanced blow-up" properties. Section 5 another secondary discussion; that is, one regarding proximity oblivious testers.

2 Preliminaries

In this section we review the definition of property testing, when specialized to graph properties in the adjacency matrix model. We also define the blow-up properties (and discuss the case of balanced blow-up).

2.1 Basic notions

For an integer n, we let $[n] \stackrel{\text{def}}{=} \{1,...,n\}$. A generic N-vertex graph is denoted by $G=([N],E)$, where $E \subseteq \{\{u,v\} : u,v \in [N]\}$ is a set of (unordered) pairs of vertices. Any set of (such) graphs that is closed under isomorphism is called a graph property. By oracle access to such a graph $G=([N],E)$ we mean oracle access to the Boolean function that answers the query $\{u,v\}$ (or rather $(u,v) \in [N]\times[N]$) with the bit 1 if and only if $\{u,v\} \in E$. At times, we look at E as a subset of $V \times V$; that is, we often identify E with $\{(u,v) : \{u,v\} \in E\}$.

Definition 2.1 (property testing for graphs in the adjacency matrix model): *A* tester *for a graph property Π is a probabilistic oracle machine that, on input parameters N and ϵ and access to an N-vertex graph $G=([N],E)$, outputs a binary verdict that satisfies the following two conditions.*

1. *If $G \in \Pi$ then the tester accepts with probability at least $2/3$.*

2. *If G is ϵ-far from Π then the tester accepts with probability at most $1/3$, where G is* ϵ-far *from Π if for every N-vertex graph $G'=([N],E') \in \Pi$ it holds that the symmetric difference between E and E' has cardinality that is greater than ϵN^2.*

If the tester accepts every graph in Π with probability 1, then we say that it has one-sided error. *A tester is called* non-adaptive *if it determines all its queries based solely on its internal coin tosses* (and the parameters N and ϵ)*; otherwise it is called* adaptive.

The query complexity of a tester is the number of queries it makes to any N-vertex graph oracle, as a function of the parameters N and ϵ. We say that a tester is efficient if it runs in time that is polynomial in its query complexity, where basic operations on elements of $[N]$ are counted at unit cost. We note that all testers presented in this paper are efficient, whereas the lower-bounds hold also for non-efficient testers.

We shall focus on properties that can be tested within query complexity that only depends on the proximity parameter, ϵ. Thus, the query-complexity upper-bounds that we state hold for any values of ϵ and N, but will be meaningful only for $\epsilon > 1/N^2$ or so. In contrast, the lower-bounds (e.g., of $\Omega(1/\epsilon)$) cannot possibly hold for $\epsilon < 1/N^2$, but they will indeed hold for any $\epsilon > N^{-\Omega(1)}$. Alternatively, one may consider the query-complexity as a function of ϵ, where for each fixed value of $\epsilon > 0$ the value of N tends to infinity.

2.2 The blow-up properties

Following the discussion in the introduction, we first define the blow-up properties that are the subject of our study.

Definition 2.2 (graph blow-up): *We say that the graph $G = ([N], E)$ is a* blow-up of the graph *$H = ([h], F)$ if there is an h-way partition $(V_1, ..., V_h)$ of the vertices of G such that for every $i, j \in [h]$ and $(u, v) \in V_i \times V_j$ it holds that $(u, v) \in E$ if and only if $(i, j) \in F$. We stress that the V_i's are not required to be of equal size and that some of them may be empty. We denote by $\mathcal{BU}(H)$* (resp., $\mathcal{BU}_N(H)$) *the set of all graphs* (resp., N-vertex graphs) *that are blow-ups of H.*

In contrast to Definition 2.2, let us briefly consider the more rigid (and popular) definition of a *balanced* blow-up.

Definition 2.3 (balanced blow-up): *We say that the graph $G = ([N], E)$ is a* balanced blow-up of the graph *$H = ([h], F)$ if there is an h-way partition $(V_1, ..., V_h)$ of the vertices of G such that the following two conditions hold:*

1. *For every $i, j \in [h]$ and $(u, v) \in V_i \times V_j$ it holds that $(u, v) \in E$ if and only if $(i, j) \in F$.*

2. *For every $i \in [h]$ it holds that $|V_i| \in \{\lfloor N/h \rfloor, \lceil N/h \rceil\}$.*

We denote by $\mathcal{BBU}(H)$ (resp., $\mathcal{BBU}_N(H)$) *the set of all graphs* (resp., N-vertex graphs) *that are balanced blow-ups of H.*

It is easy to see that, except for trivial cases (i.e., when H consists of isolated vertices), balanced blow-up cannot be tested with one-sided error and complexity that does not depend on the size of the graph. The two-sided error testing complexity of this property is $\Theta(1/\epsilon^2)$, as shown next.

Proposition 2.4 (on the complexity of testing balanced blow-up): *For every $H = ([h], F)$ such that $F \neq \emptyset$, testing the property $\mathcal{BBU}(H)$ requires $\Omega(1/\epsilon^2)$ queries even if adaptive testers of two sided error are allowed. On the other hand, for any $H = ([h], F)$, there exists a non-adaptive tester of query complexity $O(1/\epsilon^2)$* (and two-sided error) *for the property $\mathcal{BBU}(H)$.*

Proof: The lower bound follows directly from the known lower bounds on estimating the average (cf. [CEG]). Specifically, distinguishing Boolean functions defined over $[N]$ and having an average value of 0.5 from Boolean functions having an average of $0.5-\epsilon$ can be reduced to distinguishing N-vertex graphs that consist of two isolated cliques of the same size from graphs that consist of two isolated cliques of sizes $(0.5-\epsilon)\cdot N$ and $(0.5+\epsilon)\cdot N$, respectively. (Given oracle access to a function $f:[N]\to\{0,1\}$ consider the graph $G=([N],\{(u,v):f(u)=f(v)\})$.)

In describing the tester, we first assume that $H=([h],F)$ is not a blow-up of any smaller graph H'. Also, anticipating the extension to the general case, we generalize the balanced blow-up property into a proportional blow-up property. Here, for a fixed graph $H=([h],F)$ and sequence of densities $\overline{\rho}=(\rho_1,..,\rho_h)$, the graph G is a $\overline{\rho}$-blow-up of H if Definition 2.3 holds with Condition 2 replaced by $|V_i|\in\{\lfloor\rho_i N\rfloor,\lceil\rho_i N\rceil\}$. The non-adaptive tester for $\overline{\rho}$-blow-up of H, where H is not a blow-up of any smaller graph, proceeds as follows (on input a graph G):

1. Select uniformly a sample of $\widetilde{O}(1/\min_i\{\rho_i\})$ vertices, denoted B, which will be used as a basis for clustering in Step 2. Select uniformly a sample of $O(|B|/\epsilon^2)$ vertices, denoted S. Finally, select uniformly a sample of $O(h^2/\epsilon)$ vertex pairs in $S\times S$, denoted T.

2. Query all pairs $(u,v)\in(B\times S)\cup T$, and cluster the vertices in S according to their neighbors in B. That is, for every $v\in[N]$, let $\mathrm{sg}_B(v)\stackrel{\rm def}{=}\{u\in B:(u,v)\in E\}$, and, for every set $B'\subseteq B$, let $S_{B'}\stackrel{\rm def}{=}\{v\in S:\mathrm{sg}_B(v)=B'\}$.

3. If the number of non-empty sets $S_{B'}$ exceeds h, then reject. Otherwise, consider all possible 1-1 mappings from $C\stackrel{\rm def}{=}\{B':S_{B'}\neq\emptyset\}$ to $[h]$, and for each such mapping ϕ determine whether or not the following two conditions hold.

   (a) For every $B'\in C$ it holds that $|S_{B'}|=(1\pm\epsilon/2)\cdot\rho_{\phi(B')}\cdot|S|$.

   (b) For every $(u,v)\in T$ it holds that $(u,v)\in E$ if and only if $(\phi(\mathrm{sg}_B(u)),\phi(\mathrm{sg}_B(v)))\in F$,

   The test accepts if and only if there exists a mapping ϕ that satisfies both the above conditions.

The number of queries performed by the tester is $O(|B|^2/\epsilon^2)=O(1/\epsilon^2)$. We first consider what happens if G is a $\overline{\rho}$-blow-up of H. In this case, with high probability, (1) the sample B contains at least one representative from each cluster of G, and (2) for each $i\in[h]$ the sample S contains $(1\pm\epsilon/2)\cdot\rho_i\cdot|S|$ representatives of the $i^{\rm th}$ cluster. In this case, the tester accepts. We now turn to the case that $G=([N],E)$ is ϵ-far from being a $\overline{\rho}$-blow-up of H. In this case, for any choice of B, we can consider the clustering of the entire graph according to sg_B, and denote the h largest clusters by $V_1,...,V_h$ (where some of these V_i's may be empty). Letting $V\stackrel{\rm def}{=}\bigcup_{i\in[h]}V_i$, we note that if $|V|<(1-\epsilon/2)\cdot N$, then with high probability we reject at the onset of Step 3 due to seeing more than h clusters in the sample.[2] Otherwise, we consider all possible mappings of the vertices of the h largest clusters to $[h]$. For each such mapping $\psi:V\to[h]$ such that $\phi(u)=\phi(v)$ iff $u,v\in V_i$ for some i, either there exists an $i\in[h]$ such that $|V_i|\notin(1\pm\epsilon/4)\cdot\rho_i N$ or there exist at least $\epsilon N^2/4$ violating pairs (i.e., vertex pairs $(u,v)\in V\times V$ that have an edge relation in G that does not fit the edge relation of $(\psi(u),\psi(v))$ in H). In the first case, with high probability, the sample S will contain a deviating fraction of vertices from V_i, whereas in the second case, with high probability,

[2] If $|V_h|\geq(\epsilon/2h)\cdot N$, then with high probability S will contain a vertex from each V_i as well as a vertex that does not belong to V. On the other hand, if $|V_h|\leq(\epsilon/2h)\cdot N$, then with high probability S will contain $h+1$ vertices from different clusters in $[N]\setminus V$.

the sample T will hit some of these violations.[3] In either cases, with high probability, the tester will reject. This completes the treatment of the case (of $\overline{\rho}$-blow-up) of a graph $H = ([h], F)$ that is not a blow-up of any smaller graph.

Finally, suppose that $H([h], F)$ is a blow-up of some smaller graph H', and suppose that H' is minimal (i.e., it is not a blow-up of any smaller graph). Then, testing the property $\mathcal{BBU}(H)$ reduces to testing a proportional blow-up property regarding H', where the proportions are determined according to the blow-up of H' into H (and the densities are multiples of $1/h$). ■

3 The $\mathcal{BU}(H)$-Tester and its Basic Features

Recall that a tester of the type we seek (i.e., a non-adaptive tester of $\widetilde{O}(1/\epsilon)$ query complexity) cannot operate by inspecting an induced subgraph, because by [GR08, Prop. 6.2] such a subgraph will have to be induced by $\Omega(1/\epsilon)$ vertices, which would yield query complexity $\Omega(1/\epsilon^2)$. Thus, like in [GR08, Sec. 6.2], our non-adaptive tester operates by using a less straightforward querying procedure. Specifically, it does select a sample of $\widetilde{O}(1/\epsilon)$ vertices, but does *not* query all vertex pairs.

Algorithm 3.1 (testing $\mathcal{BU}(H)$, for a fixed graph $H = ([h], F)$): *On input parameters, N and ϵ, and access to an oracle $g : [N] \times [N] \to \{0, 1\}$, representing a graph $G = ([N], E)$, the algorithm sets $\ell = \log_2(1/\epsilon) + O(\log\log(1/\epsilon))$ and proceeds as follows.*

1. *For every $i \in [\ell]$, it selects uniformly a sample of $\mathrm{poly}(\ell) \cdot 2^i$ vertices, denoted T_i.*

   Denote $T = \bigcup_{i\in[\ell]} T_i$.

2. *For every $i, j \in [\ell]$ such that $i + j \leq \ell$, the algorithm queries all pairs in $T_i \times T_j$.*

3. *The algorithm accepts if and only if the answers obtained in Step 2 are consistent with some blow-up of H. That is, let $K : T \times T \to \{0, 1, *\}$ be a partial description of the subgraph of G induced by T such that $K(u, v) = g(u, v)$ if query (u, v) was made in Step 2, and otherwise $K(u, v) = *$. Then, the acceptance condition seeks a mapping $\phi : T \to [h]$ such that if $K(u, v) = 1$ then $(\phi(u), \phi(v)) \in F$ and if $K(u, v) = 0$ then $(\phi(u), \phi(v)) \notin F$.*

Indeed, at this point we ignore the computational complexity of implementing Step 3. We shall return to this issue at the end of the current section. But, first, let us note that the query complexity of Algorithm 3.1 is

$$\sum_{i,j:i+j\leq\ell} \mathrm{poly}(\ell) \cdot 2^{i+j} \;=\; \mathrm{poly}(\ell) \cdot 2^{\ell} \;=\; \widetilde{O}(1/\epsilon). \tag{1}$$

It is also clear that Algorithm 3.1 is non-adaptive and that it accept every $G \in \mathcal{BU}(H)$ with probability 1 (i.e., it has one-sided error). The bulk of this work (see Section 4) is devoted to showing that if G is ϵ-far from $\mathcal{BU}(H)$, then Algorithm 3.1 rejects it with probability at least 2/3.

Relaxing the acceptance condition of Algorithm 3.1. A straightforward implementation of Step 3 amounts to considering all $h^{|T|}$ mappings of T to $[h]$, and checking for each such mapping ϕ whether the clustering induced by ϕ fits the graph H. Relaxing the acceptance condition (used in Step 3 of Algorithm 3.1) yields a more time-efficient algorithm. Actually, the relaxed acceptance

[3]Note that a $1/h^2$ fraction of these foregoing violations can be attributed to one of $2 \cdot \binom{h}{2}$ events that correspond to the existence or non-existence of edges between some pair of clusters.

condition (defined next) seems easier to analyze than the original one. The notion of *pairwise inconsistent rows* (of K) is pivotal to this relaxed acceptance condition. (Indeed, it will be instructive to think of K as a matrix, and to view rectangular restrictions of K as sub-matrices.)

Definition 3.2 (pairwise inconsistent rows): *Let $K' : R \times C \to \{0,1,*\}$ be a sub-matrix of $K : T \times T \to \{0,1,*\}$; that is, $R, C \subseteq T$ and $K'(r,c) = K(r,c)$ for for every $(r,c) \in R \times C$. Then, the rows $r_1, r_2 \in R$ are said to be* inconsistent (wrt K') *if there exists a column $c \in C$ such that $K'(r_1,c)$ and $K'(r_2,c)$ are different Boolean values* (i.e., $K'(r_1,c), K'(r_2,c) \in \{0,1\}$ and $K'(r_1,c) \neq K'(r_2,c)$). *A set of rows of K' is called* pairwise inconsistent (wrt K') *if each pairs of rows is inconsistent* (wrt K').

Another pivotal notion, which was alluded to before, is the notion of being consistent with some blow-up of H, which we now term *H-mappability.*

Definition 3.3 (H-mappable sub-matrices): *Let $K' : R \times C \to \{0,1,*\}$ be a sub-matrix of $K : T \times T \to \{0,1,*\}$. We say that K' is H-*mappable *if there exists a mapping $\phi : R \to [h]$ such that if $K'(u,v) = 1$ then $(\phi(u), \phi(v)) \in F$ and if $K'(u,v) = 0$ then $(\phi(u), \phi(v)) \notin F$. We call such a ϕ an H-*mapping of K' (or R) to $[h]$.

Note that if K is H-mappable, then every two *inconsistent* rows of K must be mapped (by ϕ as in Definition 3.3) to different vertices of H. In particular, if a sub-matrix $K' : R \times C \to \{0,1,*\}$ of K has pairwise inconsistent rows, then any H-mapping of K to $[h]$ must be injective. Hence, if K contains more than h pairwise inconsistent rows, then K is not H-mappable.

Definition 3.4 (the relaxed acceptance condition (of Algorithm 3.1)): *The relaxed algorithm accept if and only if each set of pairwise inconsistent rows in K is H-mappable. That is, for every set R of pairwise inconsistent rows in K, we check whether the sub-matrix $K' : R \times T \to \{0,1,*\}$ is H-mappable, where the pairwise inconsistency condition mandates that this mapping of R to $[h]$ be 1-1. In particular, if K has more than h pairwise inconsistent rows, then the relaxed acceptance condition fails.*

Note that the relaxed acceptance condition can be checked by considering all s-subsets of T, for all $s \leq h+1$. For each such subset that consists of pairwise inconsitent rows, we consider all possible 1-1 mappings of this subset to $[h]$, and check consistency with respect to H. This can be performed in time $\binom{|T|}{h+1} \cdot (h!) < |T|^{h+1} = \mathrm{poly}(1/\epsilon)$, where the polynomial depends on h.

Clearly, if $G \in \mathcal{BU}(H)$, then for every $T \subseteq [N]$ it holds that the corresponding matrix K satisfies Definition 3.4. Thus, the relaxed algorithm always accepts graphs in $\mathcal{BU}(H)$. Section 4 is devoted to showing that if G is ϵ-far from $\mathcal{BU}(H)$, then the relaxed algorithm rejects with high probability.

4 The Acceptance Condition and Graphs that are far from $\mathcal{BU}(H)$

In light of the above, Theorem 1.1 follows from the fact that the relaxed version of Algorithm 3.1 (which uses the condition in Definition 3.4) rejects with very high probability any graph G that is ϵ-far from $\mathcal{BU}(H)$. This fact is established next.

Lemma 4.1 (main lemma): *Suppose that $G = ([N], E)$ is ϵ-far from $\mathcal{BU}_N(H)$, and let $T = \bigcup_{i \in [\ell]} T_i$ be selected at random as in Step 1 of Algorithm 3.1. Then, with probability at least $2/3$, there exists a set $R \subset T$ of pairwise inconsistent rows in the corresponding matrix $K : T \times T \to \{0,1,*\}$ that is not H-mappable,*

Before embarking on the actual proof of Lemma 4.1, we provide a very rough outline.

Outline of the proof of Lemma 4.1. Our very rough plan of action is to partition the selection of T (and each of its parts, i.e., $T_0, T_1, ..., T_\ell$) into $p(\ell) \stackrel{\rm def}{=} 2\ell^h$ many phases such that in the $j^{\rm th}$ phase we select at random samples $T_0^j, T_1^j, ..., T_\ell^j$ such that $|T_i^j| = \text{poly}(\ell) \cdot 2^i$. Thus, we let each T_i equal $\bigcup_{j=1}^{p(\ell)} T_i^j$, but we shall consider the queries as if they are made in phases such that in the $j^{\rm th}$ phase we only consider queries between $T^j \stackrel{\rm def}{=} \bigcup_{i\in[\ell]} T_i^j$ and $T^{[j]} \stackrel{\rm def}{=} \bigcup_{k\le j} T^k$. Letting $K^j : T^{[j]} \times T^{[j]} \to \{0, 1, *\}$ denote the partial information obtained on G in the first j phases, we consider a certain set R^j of pairwise inconsistent rows of K^j. If this set R^j is not H-mappable, then we are done. Otherwise, we show that, with high probability over the choice of the sample T^{j+1}, we obtain a new set R^{j+1} of pairwise inconsistent rows such that R^{j+1} has a higher *index* than R^j, where the indices refer to an order over sequences of length at most h over $[\ell]$. Since the number of such sequences is $\sum_{k\in[h]} \ell^k < p(\ell)$, with high probability, this process must reach a set R^j that is not H-mappable, and so we are done.

Needless to say, the crucial issue is the progress achieved in each phase; that is, the fact that at each phase j the index of the new set R^{j+1} is higher than the index of the old set R^j. Intuitively, this progress is achieved because the current (H-mappable) set R^j induces a clustering of all vertices of G that extends this H-mapping, whereas this mapping must contain many vertex pairs that violate the edge relation of H. The sample taken in the current phase (i.e., T^{j+1}) is likely to hit these violations, and this gives rise to a set R^{j+1} with higher index.

4.1 Basic notions and notations

In addition to the foregoing notations, T_i^j, T^j and $T^{[j]}$, we shall use the following notations.

- A pair (R, C) is called a j-basic pair if $C \subseteq T^{[j]}$ and $R \subseteq C$. Indeed, j-basic pairs correspond to restrictions of the sample available at phase j (i.e., $T^{[j]}$).

- The j-index of a vertex $v \in T^{[j]}$, denoted $\texttt{idx}^j(v)$, is the smallest index i such that $v \in T_i^{[j]}$, where $T_i^{[j]} \stackrel{\rm def}{=} \bigcup_{k\le j} T_i^k$. (Note that $\texttt{idx}(\cdot)$ depends on T, but this dependence is not shown in the notation.)

  A key observation is that for every $u, v \in T$, it holds that $K(u, v) = g(u, v)$ if and only if $\texttt{idx}^{p(\ell)}(u) + \texttt{idx}^{p(\ell)}(v) \le \ell$. Othewise, $K(u, v) = *$ (indicating that (u, v) was not queried by Algorithm 3.1.

  We comment that, with extremely high probability, for each j and $v \in T^{[j]}$, there exists a unique $i \in [\ell]$ and $k \in [j]$ such that $v \in T_i^k$. Thus, for any $v \in T^{[j]}$, we may assume that $\texttt{idx}^{j+1}(v) = \texttt{idx}^j(v)$.

- The indices of individual vertices in $T^{[j]}$ are the basis for defining the index of sets in $T^{[j]}$. Specifically, the j-index of a set $S \subseteq T^{[j]}$, denoted $\texttt{idx}^j(S)$, is the multi-set consisting of all values $\texttt{idx}^j(v)$ for $v \in S$. It will be instructive to consider an ordered version of this multi-set; that is, we redefine $\texttt{idx}^j(S)$ as $(i_1, ..., i_{|S|})$ such that (1) for every $k < |S|$ it holds that $i_k \ge i_{k+1}$, and (2) for every $i \in [\ell]$ it holds that $|\{k \in [|S|] : i_k = i\}| = |\{v \in S : \texttt{idx}^j(v) = i\}|$.

- We consider a natural lexicographic order over sequences, denoted $\succ$, such that for two (monotonicly non-increasing) sequences of integers, $a = (a_1, ..., a_m)$ and $b = (b_1, ..., b_n)$, it holds that $a \succ b$ if

  - either there exists $i \le \min(n, m)$ such that $(a_1, ..., a_{i-1}) = (b_1, ..., b_{i-1})$ and $a_i > b_i$.

– or $m > n$ and $(a_1, ..., a_n) = (b_1, ..., b_n)$.

Note that $\succ$ is a total order on the set of monotonicly non-increasing (finite) sequences of integers.

As hinted in the overview, a key notion in our analysis is the notion of a clustering of the vertices of G that is induced by an H-mapping of some small subset of vertices. Actually, the clustering is induced by a partial knowledge sub-matrix $K' : R \times C \to \{0,1,*\}$ as follows.

Definition 4.2 (the clustering induced by K'): *Let $K' : R \times C \to \{0,1,*\}$ be a sub-matrix of $K : T \times T \to \{0,1,*\}$ such that K'* has pairwise inconsistent rows. *Then, for every $r \in R$, we denote by $V_r(K')$ the set of vertices $v \in [N]$ that are consistent with row r in K'. That is,*

$$V_r(K') \stackrel{\rm def}{=} \{v \in [N] : (\forall c \in C)\ g(v,c) \cong K'(r,c)\} \tag{2}$$

where, for $\sigma, \tau \in \{0,1,*\}$, we write $\sigma \cong \tau$ if either $\sigma = \tau$ or $\sigma = *$ or $\tau = *$. The vertices that are inconsistent with all rows, are placed in the leftover set $L(K') \stackrel{\rm def}{=} [N] \setminus \bigcup_{r\in R} V_r(K')$.

Indeed, rows $r_1, r_2 \in R$ are inconsistent wrt K' (as per Definition 3.2) if there exists a column $c \in C$ such that $K'(r_1,c) \ncong K'(r_2,c)$ (which means that $K'(r_1,c)$ and $K'(r_2,c)$ are both in $\{0,1\}$ but are different). Thus, the hypothesis that the rows of K' are pairwise inconsistent implies that the sets in Eq. (2) are disjoint. Hence, the clustering in Definition 4.2 is indeed a partition of the vertex set of G (since $v \in L(K')$ if for every $r \in R$ there exists $c \in C$ such that $g(v,c) \ncong K'(r,c)$). This motivates our focus on sub-matrices having pairwise inconsistent rows. The following definition adds a requirement (regarding such sub-matrices) that refers to the relation between the index of row r and the density of the corresponding set $V_r(K')$.

Definition 4.3 (nice pairs): *Let (R,C) be a j-basic pair and $K' : R \times C \to \{0,1,*\}$ be the corresponding sub-matrix of K. We say that (R,C) is a j-*nice pair* if the following two conditions hold.*

1. *R are pairwise inconsistent with respect to K'.*
2. *For every $r \in R$ it holds that $\mathtt{ind}^j(r) \le \rho(V_r(K')) + 1$, where $\rho(S) \stackrel{\rm def}{=} \lceil \log(N/|S|) \rceil$.*

As a sanity check, suppose that $r \in R$ was selected in phase j (i.e., $r \in T^j$). Then, it is very likely that r (or some other member of $V_r(K')$) is selected in $T^j_{\rho(V_r(K'))-1}$, because $T^j_{\rho(V_r(K'))-1}$ is a random set of cardinality $\mathrm{poly}(\ell) \cdot 2^{\rho(V_r(K'))-1} = \mathrm{poly}(\ell) \cdot N/|V_r(K')|$.

For each phase j, we shall show the existence of a j-nice pair. Furthermore, we shall show that the corresponding set of rows has a higher index than all sets of rows associated with previous phases. The furthermore claim is the crux of the analysis, and is captured by the Progress Lemma presented in Section 4.2. But let us first establish the mere existence of j-nice pairs. Indeed, for every $j \ge 1$, we may pick an arbitrary $r \in T_1^1$, and consider the j-nice pair $(\{r\},\{r\})$, while noting that $\mathtt{idx}^1(r) = 1$ and $\rho(V_r(K') \ge 0$ (where $K' : \{r\} \times \{r\} \to \{0,1,*\}$).

4.2 The Progress Lemma

Recall that $G = ([N], E)$ is ϵ-far from $\mathcal{BU}(H)$, where $H = ([h], F)$. Furthermore, we consider the partial view $K : T \times T \to \{0, 1, *\}$ obtained by Algorithm 3.1, where $T = \bigcup_{i\in[\ell],j\in[p(\ell)]} T_i^j$ is the random sample is selected. Throughout the rest of this section, we say that an event has negligible probability if it occurs with probability that vanishes faster than any polynomial in ϵ. Since we shall consider only poly(ℓ) many events, we can safely ignore these negligible probabilities.[4] We say that an event occurs with overwhelmingly high probability if the probability that it does *not* occur is negligible.

Lemma 4.4 (Progress Lemma): *Let (R, C) be a j-nice pair and $K' : R \times C \to \{0, 1, *\}$ be the corresponding sub-matrix of K. If K' is H-mappable then, with overwhelmingly high probability over the choice of T^{j+1}, there exists a $(j + 1)$-nice pair (R', C') such that $\mathtt{ind}^{j+1}(R') \succ \mathtt{ind}^j(R)$.*

Recalling that a (trivial) 1-nice pair always exists and that the number of possible indices is smaller than $p(\ell)$, we conclude that, with overwhelmingly high probability (over the choice of T), there exists a $j < p(\ell)$ and a j-nice pair that is not H-mappable. Lemma 4.1 follows. Thus, all that remains is proving Lemma 4.4, which we undertake next.

Proof: We consider the partition induced by K', as per Definition 4.2, and consider two cases regarding the size of $L \stackrel{\rm def}{=} L(K')$:

Case 1: $\rho(L) \leq \ell$. In this case (i.e., $|L| \geq 2^{-\ell} \cdot N$), with overwhelmingly high probability, the sample T^{j+1} contains a vertex $u \in L(K')$. Using this u, we shall obtain a $(j + 1)$-nice pair with a set of rows that has a higher index than R. Intuitively, since $(g(u, c))_{c\in C}$ is inconsistent with all rows of K', we may add u as a row to K' while possibly omitting rows of K' that are consistent with $(K(u, c))_{c\in C}$ (see below), obtaining a sub-matrix that has a larger index (than the index of K'). The detailed analysis of this case is presented in Claim 4.4.2.

Case 2: $\rho(L) > \ell$. In this case (i.e., $|L| < 2^{-\ell} \cdot N < \epsilon N/2$), the partition induced by $(V_r(K'))_{r\in R}$ contains many pairs that violate the edge relation of H, since the number of pairs adjacent at L is smaller than $\epsilon N^2/2$. We shall show that, with overwhelmingly high probability, the sample T^{j+1} contains a vertex w such that augmenting K' with the column corresponding to w yields a sub-matrix K'' such that $\rho(L(K'')) < \ell$. Intuitively, pairs of vertices in $V(K') \stackrel{\rm def}{=} \bigcup_{r\in R} V_r(K')$ that violate the edge relation of H, yield vertices w that effectively shrink $V(K')$ in the sense that adding w as a column to K' moves many vertices from $V(K')$ to $L(K'')$. In particular, we shall show that $|L(K'')| = \Omega(\epsilon N/\ell)$, which means that $\rho(L(K'')) < \log_2(O(\ell)/\epsilon) < \ell$. At this point we may proceeds as in Case 1. (Formally, in this case, the $j+1$st phase is partitioned into two sub-phases, where in each sub-phase we use half of each of the samples T_i^{j+1}.) The detailed analysis of this case is presented in Claim 4.4.3.

Our analysis of the two cases combines straightforward probabilistic arguments with manipulations of sub-matrices. The latter manipulations include adding rows and columns and truncating the sub-matrix so as to leave only rows that have an index that is lower-bounded by some value. It is thus instructive to discuss these three operations first.

[4] In fact, it would have sufficed to define as negligible any probability that vanishes faster than any polynomial in $1/\ell$ (i.e., faster than any polylogarithmic function of ϵ).

Adding an arbitrary column from T^{j+1}. Suppose that (R, C) is j-nice with a corresponding sub-matrix K'. Then, adding any column $v \in T^{j+1}$ to K' results in a sub-matrix K'' such that the corresponding pair $(R, C \cup \{v\})$ is $(j+1)$-nice. Clearly, adding a column may only add inconsistencies, and so the pairwise inconsistency condition of K' is preserved. For any $r \in R$, the densities of $V_r(\cdot)$ may only drop when moving from K' to K'', and so $\mathtt{ind}^j(r) \le \rho(V_r(K')) + 1$ implies $\mathtt{ind}^{j+1}(r) \le \rho(V_r(K'')) + 1$.

Adding a row that belongs to $L(K') \cap T^{j+1}_{\rho(L(K'))}$. It is tempting to think that if (R, C) is j-nice, then adding any row $v \in T^{j+1}_{\rho(L(K'))} \cap L(K') \cap C$ to K' results in a sub-matrix K'' such that the corresponding pair $(R \cup \{v\}, C)$ is $(j+1)$-nice. It is true that $\mathtt{ind}^{j+1}(r) \le \rho(V_r(K'')) + 1$ holds for each row r, including the added row v (because $\mathtt{ind}^{j+1}(v) = \rho(L(K'))$ and $\rho(V_v(K'')) \ge \rho(L(K'))$, since $V_v(K'') \subseteq L(K')$). However, although for every $r \in R$ there exists $c \in C$ such that $g(v, c) \ncong K'(r, c)$ (since $v \notin V_r(K')$), it not necessarily the case that the row v in K is inconsistent with all rows in K' (i.e., it may be the case that, for some $r \in R$ and each $c \in C$, it holds that $K(v, c) \cong K'(r, c)$, since $K(v, c) \in \{g(v, c), *\}$ and $* \cong K'(r, c)$). Coping with this problem, which arises from the fact that K may have 8-values, leads us to introduce the following truncation operator.

Truncating at an added row. Suppose that (R, C) is j-nice with a corresponding sub-matrix K', and let $v \in L(K') \cap T^{j+1}_{\rho(L(K'))}$. Then, consider first adding v as a new row and column to K', and then leaving in the resulting sub-matrix only the rows that have a $(j+1)$-index that is at least as large as the one of v (i.e., row r remains if and only if $\mathtt{ind}^{j+1}(r) \ge \mathtt{ind}^{j+1}(v)$). We claim that these rows are pairwise inconsistent, and thus the resulting sub-matrix is $(j+1)$-nice.

It suffices to prove that the new row v (of K) is inconsistent with any row that was left from K'; that is, fixing any $r \in R$ such that $\mathtt{ind}^{j+1}(r) \ge \mathtt{ind}^{j+1}(v)$, we claim that there exists $c \in C$ such that $K(v, c) \ncong K'(r, c)$. Since $v \in L(K')$, we know that there exists $c \in C$ such that $g(v, c) \ncong K'(r, c)$, which implies that $K'(r, c) \in \{0, 1\}$, which in turn implies $\mathtt{ind}^j(r) + \mathtt{ind}^j(c) \le \ell$ (by definition of K). Now, using $\mathtt{ind}^{j+1}(v) \le \mathtt{ind}^{j+1}(r) \le \mathtt{ind}^j(r)$, we get $\mathtt{ind}^{j+1}(v) + \mathtt{ind}^j(c) \le \ell$, which implies that $K(v, c) = g(v, c)$. Recalling that $g(v, c) \ncong K'(r, c)$, we obtain $K(v, c) \ncong K'(r, c)$, and the claim follows.

Note that the truncation of $K' : R \times C \to \{0, 1, *\}$ at the added row v always contains the new row v, and that it may result in $|R| + 1$ rows (i.e., no "real truncation"). Another key feature of the truncation-at-an-added-row operation is that it yields a set of rows with an index larger than the index of R.

Claim 4.4.1 (the effect of truncation): *Suppose that (R, C) is j-nice with a corresponding sub-matrix K', and let $v \in L(K') \cap T^{j+1}_{\rho(L(K'))}$. Then, truncating the sub-matrix that corresponds to $(R \cup \{v\}, C \cup \{v\})$ at row v yields a $(j+1)$-nice pair with a row set having an index larger than $\mathtt{ind}^j(R)$.*

Proof: The first part of this claim was already established above. Denoting the resulting set of rows by R', we need to prove that $\mathtt{ind}^{j+1}(R') \succ \mathtt{ind}^j(R)$. If $R' = R \cup \{u\}$ then the claim is trivial, and so we consider the case that $\mathtt{ind}^{j+1}(R') = (i_1, ..., i_t)$, where $t \le |R|$ and $i_t = \mathtt{ind}^{j+1}(v)$. This means that a non-trivial truncating took place, and that all omitted rows had index smaller than i_t, which implies that $(i_1, ..., i_t) \succ \mathtt{ind}^{j+1}(R)$ (because $\mathtt{ind}^{j+1}(R) = (i_1, ..., i_{t-1}, d_t, ..., d_{|R|})$ with $d_t < i_t$). $\square$

Claim 4.4.2 (case 1): *Suppose that (R, C) is j-nice and that $\rho(L) \le \ell$, where $L = L(K')$. Then, with overwhelmingly high probability* (over the choice of $T^{j+1}_{\rho(L(K'))}$), *the sample $T^{j+1}_{\rho(L(K'))}$ contains a vertex $u \in L(K')$ such that adding u to K'* (both as a row and a column) *and truncating the resulting sub-matrix at row u yields a $(j+1)$-nice pair (R', C') such that $\mathtt{ind}^{j+1}(R') \succ \mathtt{ind}^{j}(R)$.*

Proof: With overwhelmingly high probability, the sample $T^{j+1}_{\rho(L(K'))}$ contains a vertex $u \in L(K')$, while using any such vertex yields the desired result (due to Claim 4.4.1). $\square$

Claim 4.4.3 (case 2): *Suppose that (R, C) is j-nice and that the corresponding sub-matrix K' is H-mappable. Further suppose that $\rho(L) > \ell$, where $L = L(K')$. Then, with overwhelmingly high probability* (over the choice of T^{j+1}), *the sample T^{j+1} contains a vertex w such that adding the column w to K' yields a $(j+1)$-nice pair $(R, C \cup \{w\})$ such that the corresponding sub-matrix K'' satisfies $\rho(L(K'')) \le \ell$*

Proof: We combine the hypothesis that G is ϵ-far from $\mathcal{BU}(H)$ with the hypothesis that K' is H-mappable, and denote the corresponding H-mapping by $\phi : R \to [h]$. Extending this mapping to $V(K') \stackrel{\rm def}{=} \bigcup_{r \in R} V_r(K')$ such that $\phi(v) = \phi(r)$ for every $v \in V_r(K')$, and using the hypothesis that $|L(K')| < 2^{-\ell} N < \epsilon N/2$, we conclude that there are at least $\epsilon N^2/2$ vertex pairs that violate the edge relation of H (i.e., pairs $(u, v) \in V(K') \times V(K')$ such that $(u, v) \in E$ iff $(\phi(u), \phi(v)) \notin F$). Actually, we should consider all $h!$ possible injections (from R to $[h]$), and apply the argument to each of them, but this only increases the error probability by a factor of $h!$. These violations can be of one of the following two types.

1. Edges $(u, v) \in E$ such that $(\phi(u), \phi(v)) \notin F$. If the number of such pairs exceeds $\epsilon N^2/4$, then we select a pair $(r, s) \in R \times R$ such that there exist at least $\epsilon N^2/4h^2$ pairs $(u, v) \in E$ for which $(\phi(u), \phi(v)) = (\phi(r), \phi(s)) \notin F$.

2. Non-edges $(u, v) \notin E$ such that $(\phi(u), \phi(v)) \in F$. If the number of such pairs exceeds $\epsilon N^2/4$, then we select a pair $(r, s) \in R \times R$ such that there exist at least $\epsilon N^2/4h^2$ pairs $(u, v) \notin E$ for which $(\phi(u), \phi(v)) = (\phi(r), \phi(s)) \in F$.

Fixing (r, s) as above we have at least $\epsilon N^2/4h^2$ violating pairs in $V_r(K') \times V_s(K')$. Next, we select an integer $m \in [\ell]$ such that there exists a set $W \subseteq V_r(K')$ of cardinality $2^{-m} \cdot N$ and every $w \in W$ participates in at least $\epsilon 2^m N/4h^2\ell > 2^{-(\ell-m-3)} \cdot N$ violating pairs (with vertices of $V_s(K')$). Clearly, $\rho(W) = m$, and so with overwhelmingly high probability T^{j+1}_m contains a vertex $w \in W$. Adding any such w as a column to K', we obtain a sub-matrix K'' and claim that $\rho(L(K'')) \le \ell - m$. Specifically, we show that every $u \in V_s(K')$ such that (u, w) is a violating pair must be in $L(K'')$, and recall that the number of such violating pairs in which w participates exceeds $2^{-(\ell-m-3)} \cdot N$.

Lastly, letting U^w denote the set of all $u \in V_s(K')$ such that (u, w) is a violating pair, we prove that $U^w \subseteq L(K'')$. Let u be an arbitrary vertex in $V_s(K')$ (and recall that $w \in V_r(K')$).

1. We first note that $\mathtt{ind}^j(r) \le \rho(V_r(K')) + 1$ (by the nicety condition), whereas $\rho(V_r(K')) \le \rho(W) = m$. Similarly, $\mathtt{ind}^j(s) \le \rho(V_s(K')) + 1$, whereas $\rho(V_s(K')) \le \rho(U^w) \le \ell - m - 3$ (since $V_s(K') \supseteq U^w$ and $|U^w| > 2^{-(\ell-m-3)} \cdot N$).

2. Combining the two foregoing facts, we conclude that $\mathtt{ind}^j(r) + \mathtt{ind}^j(s) \le \ell$, which implies that $K'(r, s) = g(r, s)$.

3. Since $w \in V_r(K')$, it must be that $g(w, s) \cong K'(r, s)$, which implies $g(w, s) = g(r, s)$ (when combined with $K'(r, s) = g(r, s)$). Since ϕ is an H-mapping it must be that $g(s, w) = g(s, r)$ fits the edge relation of $(\phi(s), \phi(w)) = (\phi(s), \phi(r))$ with respect to H.

4. On the other hand, if (u,w) is a violating pair, then $g(u,w)$ does not fit the edge relation of $(\phi(u),\phi(w)) = (\phi(s),\phi(r))$ with respect to H.

5. Combining Items 3 and 4, we infer that $g(u,w) \neq g(s,w)$, which implies $g(u,w) \not\cong K''(s,w)$ (because $K''(s,w) = g(s,w)$ by virtue of $\mathtt{ind}^{j+1}(s) + \mathtt{ind}^{j+1}(w) \leq (\ell - m - 2) + m < \ell$, where $w \in T_m^{j+1}$ by the hypothesis). Thus, u is not in $V_s(K'')$, although it is in $V_s(K')$.

6. We observe that, for every $r \in R \setminus \{s\}$, vertex $u \in V_s(K')$ is not in $V_r(K'') \subseteq V_r(K')$, since the rows of K' are pairwise inconsistent.

7. Combining Items 5 and 6, we conclude that $u \notin \bigcup_{r \in R} V_r(K'')$, and hence $u \in L(K'')$.

The claim follows (since $|L(K'')| \geq |U^w| \geq 2^{-(\ell-m-3)} \cdot N > 2^{-\ell}N$). □

Completing the proof of Lemma 4.4. In accordance with the motivating discussion, we now complete the proof of the lemma by using the two latter claims. Specifically, if Case 1 holds (i.e., $\rho(L(K')) \leq \ell$), then we invoke Claim 4.4.2 anre are done. Otherwise, Case 2 holds (i.e., $\rho(L(K')) > \ell$), and we take the following two steps. Recall that, as stated in the beginning of the proof, in this case (i.e., Case 2) we partition the sample T^{j+1} into two parts, and use a different part in each step. In the first step we apply Claim 4.4.3 to the first part, and get into Case 1; that is, we obtain a new K' such that $\rho(L(K')) \leq \ell$. Next, in the second step, we apply Claim 4.4.3 to the resulting K' and the second part of the sample, and are done. ■

5 Proximity Oblivious Testing of Blow-Up

In this section we derive, for every fixed graph H, a constant-query proximity oblivious tester of $\mathcal{BU}(H)$. That is, we refer to the following definition of [GR09], when specialized to the dense graph model.

Definition 5.1 (proximity oblivious testing for graphs in the adjacency matrix model): *A* proximity oblivious tester *for a graph property Π is a probabilistic oracle machine that, on input parameter N and access to an N-vertex graph $G = ([N], E)$, outputs a binary verdict that satisfies the following two conditions.*

1. *If $G \in \Pi$, then the tester accepts with probability 1.*
2. *There exists a monotone function $\rho : (0,1] \to (0,1]$ such that, for every graph $G = ([N], E) \notin \Pi$, it holds that the tester rejects G with probability at least $\rho(\delta_\Pi(G))$, where $\delta_\Pi(G)$ denotes the* (relative) *distance of G from the set of N-vertex graphs that are in Π.*

The function ρ is called the detection probability *of the tester.*

Combining Lemma 4.1 and the ideas underlying [GR09, Thm. 6.3], we obtain.

Theorem 5.2 *For every fixed graph $H = ([h], F)$, there exists a $O(h^2)$-query proximity oblivious tester of $\mathcal{BU}(H)$. Furthermore, the tester has detection probability $\rho(\epsilon) = \epsilon^{O(h)}$.*

This extends the result of [GR09, Prob. 4.11], which corresponds to the special case in which H is a h-vertex clique. We also mention that, for constant-query proximity oblivious testers of $\mathcal{BU}(H)$, detection probability of the form $\rho(\epsilon) = \epsilon^{\Omega(h)}$ is essential (cf. [GR09, Prob. 4.3]).

Proof: While a direct application of [GR09, Thm. 6.3] would yield a detection bound of $\rho(\epsilon) = \epsilon^{O(h^2)}$, we obtain a quantative improvement by using a version of [GR09, Thm. 6.3] that is specialized to the dense graph model. This version refers to *any graph property* Π *having a standard tester* T (of error probability 1/3) *that satisfies the following three conditions*:

1. T *is non-adaptive*;
2. *for a monotonically non-decreasing* $\nu : (0, 1] \to \mathbb{N}$, *on proximity parameter* ϵ, *the queries of* T *refer to at most* $\nu(\epsilon)$ *vertices*; *and*
3. *for some fixed* $s \in \mathbb{N}$, *the tester* T *rejects if and only if it sees a partial view of some* s*-vertex subgraph that cannot occur in any graph in* Π. (Such a partial view is called a witness for non-membership.)

In such a case, Π *has an* $\binom{s}{2}$*-query proximity-oblivious tester with detection probability at least* $\rho(\epsilon) = \Omega(\epsilon/\nu(\epsilon/2)^s)$. We mention that a direct application of [GR09, Thm. 6.3] would have yielded a detection bound of $\rho(\epsilon) = \Omega(\epsilon/q(\epsilon/2)^{\binom{s}{2}})$, where $q < \nu^2$ denotes the query complexity of the original tester.

The foregoing claim is easily proved by following the ideas that underly the proof of [GR09, Thm. 6.3]. Specifically, the proximity oblivious tester select $i \in \{1, ..., \lceil \log_2 N \rceil\}$ with probability 2^{-i}, invokes the query-generator procedure of T on input ((alleged) proximity parameter) 2^{-i}, selects uniformly s vertices among those that appear in the generated queries, makes (only) the corresponding $\binom{s}{2}$ queries, and accept if and only if the induced subgraph is not a witness for non-membership. Clearly, the resulting tester rejects any graph that is 2^{-i}-far from Π with probability at least $2^{-i} \cdot \frac{2}{3} \cdot \binom{\mu(2^{-i})}{s}^{-1}$.

It remains to show that, when applied to $\Pi = \mathcal{BU}(H)$, the (non-adaptive) tester in Algorithm 3.1 (when using the relaxed condition of Definition 3.4) rejects based on a witness for non-membership that contains $O(h)$ vertices. Essentially, this holds since the condition in Definition 3.4 refers to a set of at most $h + 1$ pairwise inconsistent rows that are not H-mappable, whereas (as shown next) only $n - 1$ columns are required in order to establish that n rows are pairwise inconsistent. Thus, it suffices to augment the set of rows R by at most $|R| - 1$ additional vertices, and derive a witness for non-membership that contains at most $2h + 1$ vertices.

Lastly, we prove that $n - 1$ columns suffice for establishing the fact that n rows are pairwise inconsistent. Starting with a row r of the largest index, we pick an arbitrary column that witnesses the inconsistence of row r with some other row r'. This column c partitions the set of rows to two non-trivial sets: the set of rows having the same value as r on column c, and the set of rows having the opposite value on this column. (Note that all rows have a binary value on column c, since we started with a row r of largest index.) The process continues, separately, with each of these two sets, and the key observation is that each split requires only one (possibly new) column. ■

6 Conclusions

We have shown a non-adaptive tester of query complexity $\widetilde{O}(1/\epsilon)$ for $\mathcal{BU}(H)$. The degree of the polynomial in the polylogarithmic factor that is hidden in the $\widetilde{O}()$ notation is $h + O(1)$, where h is the number of vertices in H. We wonder whether the query complexity can be reduced to $p(h \log(1/\epsilon))) \cdot \epsilon^{-1}$, where p is a fixed polynomial. We mention that such a dependence on h was obtained in [GR08, Sec. 6.2] for the special case in which H is an h-clique. Furthermore, we wonder whether non-adaptive testing of $\mathcal{BU}(H)$ is possible in query complexity $\mathrm{poly}(h) \cdot \epsilon^{-1}$. We mention that such a result is only known for $h = 2$ (cf. [GR08, Sec. 6.1]), whereas an *adaptive* tester of query complexity $O(h^2/\epsilon)$ is known (cf. [A, Sec. 4]).

Acknowledgments

We are grateful to Dana Ron for comments regarding a previous version of this work.

References

[AFKS] N. Alon, E. Fischer, M. Krivelevich and M. Szegedy. Efficient Testing of Large Graphs. *Combinatorica*, Vol. 20, pages 451–476, 2000.

[AFNS] N. Alon, E. Fischer, I. Newman, and A. Shapira. A Combinatorial Characterization of the Testable Graph Properties: It's All About Regularity. In *38th STOC*, pages 251–260, 2006.

[AS] N. Alon and A. Shapira. A Characterization of Easily Testable Induced Subgraphs. *Combinatorics Probability and Computing*, 15:791–805, 2006.

[A] L. Avigad. On the Lowest Level of Query Complexity in Testing Graph Properties. Master thesis, Weizmann Institute of Science, December 2009.

[CEG] R. Canetti, G. Even and O. Goldreich. Lower Bounds for Sampling Algorithms for Estimating the Average. *IPL*, Vol. 53, pages 17–25, 1995.

[GGR] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, pages 653–750, July 1998.

[GKNR] O. Goldreich, M. Krivelevich, I. Newman, and E. Rozenberg. Hierarchy Theorems for Property Testing. *ECCC*, TR08-097, 2008. Extended abstract in the proceedings of *RANDOM'09*.

[GR08] O. Goldreich and D. Ron. Algorithmic Aspects of Property Testing in the Dense Graphs Model. *ECCC*, TR08-039, 2008.

[GR09] O. Goldreich and D. Ron. On Proximity Oblivious Testing. *ECCC*, TR08-041, 2008. Extended abstract in the proceedings of the *41st STOC*, 2009.

[GT] O. Goldreich and L. Trevisan. Three theorems regarding testing graph properties. *Random Structures and Algorithms*, Vol. 23 (1), pages 23–57, August 2003.

[RS] R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2), pages 252–271, 1996.